NPS ARCHIVE
1997.09
MAYS, E.

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY CA 93943-5101

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

SHEPHERD ROTARY VEHICLE: MULTIVARIATE MOTION CONTROL AND PLANNING

by

Edward J. Mays
and
Ferdinand A. Reid

September 1997

Thesis Advisor: Yutaka Kanayama

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1997	Master's Thesis

4. TITLE AND SUBTITLE
SHEPHERD ROTARY VEHICLE: MULTIVARIATE MOTION CONTROL AND PLANNING

5. FUNDING NUMBERS

6. AUTHOR(S)
Mays, Edward J., and Reid, Ferdinand, A.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

Millions of acres of the US formerly used defense sites (FUDS) are contaminated with unexploded ordnance (UXO) as a result of past military use. The process of returning the land to the civilian sector is sensitive, intensive, and costly (e.g., millions of dollars, and the loss of human life). Hence "clearing" (i.e., site remediation, range clearance, and explosive ordnance disposal) UXO's from FUDS is a complex problem. Existing clearing methods are inaccurate, dangerous, and labor intensive. This thesis shows that through robotics technology (e.g., "Shepherd" rotary vehicle with three degrees of freedom) and the use of advanced computer technology it is possible to make clearing tasks safer, more cost-effective, and more efficient. An over arching hardware and software architecture was developed for Shepherd (including a self-contained on-board computer system). The software system was developed for timer control, motion control, user interface, and an operating kernel. The hardware and software organization, structure, and interaction provide the framework for real-time control. This research included the use of encoders, digital boards, and a counter board; required the handling of interrupts, electric motor manipulation by servomotor controllers, and communication using RS232 and VMEbus technology. The kinematics algorithms and a real-time operating kernel were implemented using the C language. "Shepherd" research has laid the foundation for the flexible, robust, and precise motion needed for UXO clearing.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Unexploded Ordnance, Artificial Neural Networks	305
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

Approved for public release; distribution is unlimited

SHEPHERD ROTARY VEHICLE: MULTIVARIATE MOTION CONTROL AND PLANNING

Edward J. Mays
Major, United States Marine Corps
B.S., University of Florida, 1984
M.S., University of Southern California, 1994

Ferdinand A. Reid
Lieutenant, United States Navy
B.B.A., Georgia State University, 1990

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 1997

NPS Archive
1997.09
Mays, E

THESIS
M39796
c.1

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY CA 93943-5101

ABSTRACT

Millions of acres of the US formerly used defense sites (FUDS) are contaminated with unexploded ordnance (UXO) as a result of past military use. The process of returning the land to the civilian sector is sensitive, intensive, and costly (e.g., millions of dollars, and the loss of human life). Hence "clearing" (i.e., site remediation, range clearance, and explosive ordnance disposal) UXO's from FUDS is a complex problem. Existing clearing methods are inaccurate, dangerous, and labor intensive. This thesis shows that through robotics technology (e.g., "Shepherd" rotary vehicle with three degrees of freedom) and the use of advanced computer technology it is possible to make clearing tasks safer, more cost-effective, and more efficient. An over arching hardware and software architecture was developed for Shepherd (including a self-contained on-board computer system). The software system was developed for timer control, motion control, user interface, and an operating kernel. The hardware and software organization, structure, and interaction provide the framework for real-time control. This research included the use of encoders, digital boards, and a counter board; required the handling of interrupts, electric motor manipulation by servomotor controllers, and communication using RS232 and VMEbus technology. The kinematics algorithms and a real-time operating kernel were implemented using the C language. "Shepherd" research has laid the foundation for the flexible, robust, and precise motion needed for UXO clearing.

TABLE OF CONTENTS

I. INTRODUCTION 1
A. MOTIVATION AND BACKGROUND 1
B. OBJECTIVES 1
C. ORGANIZATION 4
II. SHEPHERD SYSTEM DESIGN 5
A. SYSTEM OVERVIEW 5
B. MOTION MODES 6
III. SHEPHERD SYSTEM HARDWARE 7
A. OVERVIEW 7
B. THE MOBILE PLATFORM 7
1. Motors and Encoders 8
2. ServoController 10
3. Gears 10
4. Wheels and Tires 12
5. Power Supply System 13
C. SHEPHERD ON-BOARD COMPUTER SYSTEM 14
1. Taurus Board 16
2. Digital to Analog Board 18
3. Digital Input Board 20
4. Digital Output Board 20
5. Counter Board 22
IV. SHEPHERD SOFTWARE SYSTEM 23
A. OVERVIEW 23
B. SOFTWARE ENVIRONMENT 23
1. Bug Monitor 23
2. S-Records 24
3. Software Development System 26
C. SHEPHERD REALTIME KERNEL (SRK) ARCHITECTURE 29
1. Timer Control 29
2. Motion Control 30
3. User Interface 31
V. EXPERIMENTAL RESULTS ON MOTION CONTROL 33
A. OVERVIEW 33
B. WHEEL DRIVING 34
1. Developing Driving Constants 34
2. Measuring Wheel Speeds 37
3. System Controls 40
C. WHEEL STEERING 49
1. Developing Steering Constants 49
2. Measuring Wheel Rate of Turn 49
3. Steering Feedback 50
4. Wheel Testing 52
VI. MOTION MODES 53
A. OVERVIEW 54
B. "TORNADO" MOTION 56
C. JOYSTICK CONTROLLED MOTION 59
D. SEARCHING MOTION 62
VII. CONCLUSIONS 69
A. SUMMARY 69

B. LESSONS LEARNED 70
C. RECOMMENDATIONS FOR FUTURE RESEARCH 71
APPENDIX A: SOURCE CODE (MAKEFILE) 73
APPENDIX B: SOURCE CODE (SHEPHERD.C) 77
APPENDIX C: SOURCE CODE (USER.C) 81
APPENDIX D: SOURCE CODE (MOVEMENT.C) 103
APPENDIX E: SOURCE CODE (MOTOR.C) 131
APPENDIX F: SOURCE CODE (TIMER.C) 165
APPENDIX G: SOURCE CODE (MATH.C) 171
APPENDIX H: SOURCE CODE (UTILS.C) 185
APPENDIX I: SOURCE CODE (SERIAL.C) 201
APPENDIX J: SOURCE CODE (CONSOLIDATED HEADER FILES) 211
APPENDIX K: SHEPHERD OPERATING MANUAL 239
APPENDIX L: SENSING SIMULATION DATA 253
APPENDIX M: SENSING SIMULATION CODE 265
APPENDIX N: INPUT VS. OUTPUT VELOCITY 283
APPENDIX O: INPUT VS OUTPUT STEERING RATES 287
LIST OF REFERENCES 291
INITIAL DISTRIBUTION LIST 293

ACKNOWLEDGEMENTS

This research was possible due to the gargantuan efforts of many people. Most notably is that of our thesis advisor, second reader, and Shepherd team members. Professor Kanayama's mentorship was truly instrumental to our in-depth understanding of kinematics and real-time systems. Professor Yun's ability to translate control systems theory into easily digestible layman's terms was invaluable. Without the efforts of team members Mike Williams and Thorsten Leonardi getting the project off the ground would have been nearly impossible—Mike's continuing maintenance efforts are a key factor in the "Shepherd" projects success.

We would also like to thank the faculty and staff members in the Computer Science Department, at the Naval Postgraduate School. Professor's Hensgen, Kidd, Lundy, and McGhee enhanced and broadened our knowledge base—even modified our Weltanschauung. Professor Luqi and the dining (hungry) philosophers provided a wonderful support network, helping us to focus on our goals. We also wish to thank the military faculty, Major Ludlow and Commander Holden for helping to ensure that our education was a value added prospect for the Department of Defense. Mrs. Valerie Brooks and Mrs. Cindy Holden should both be awarded medals for suffering through all the demos and constant noise from the robot.

Mrs. Jean Brennan and Mrs. Alice Lee of the CS curricular deserve great thanks for being a constant font of information, and for continuing to jump through all those last minute hoops with a smile.

Finally, We are grateful and indebted to our families their patience, understanding, and support throughout this research and our studies. Most of all for our life, health, and existence—Thanks be to the God of all creation!

I. INTRODUCTION

A. MOTIVATION AND BACKGROUND

Land mines are an inexpensive and effective defensive means in wars. The problem with land mines is that they remain to be a threat when wars are over. International efforts are being made to ensure that land mines deployed in the future are equipped with a time-out device, and mine locations are properly recorded. While such a treaty may provide relief in the future, millions of land mines were planted all over the world as a result of wars and regional conflicts in the past.

There are about 110 million land mines scattered around the world in more than 60 countries --- most of them in the Third World [Ref. 1, 2, 3, 4, and 5]. These land mines kill about 10,000 and injure another 20,000 people (many of them are children) every year. Moreover, there are millions of acres of the US formerly used defense sites (FUDS) that are contaminated with unexploded ordnance (UXO) as a result of military testing and training in the past [Ref. 6]. The contaminated land must be cleared inch by inch before transferring to civilian use. The difficulties of these clearing missions are in the variety of the objects to be identified and the diversity of the environments that are contaminated.

B. OBJECTIVES

As the military continues to downsize, the process of turning the land over to the civilian sector is sensitive, intensive, and costly. The aforementioned costs are both monetary and in some instance the loss of human life. One of the most complex problems is the clearing of UXO's from the FUDS.

The Department of Defense (DOD) has recently approved two organizational structures to confront the challenge of UXO remediation and wide-area de-mining. The objective of the first committee is to develop fully coordinated requirements driven research and development program for countermine, de-mining, site remediation, range clearance, and explosive ordnance disposal. Within the first committee there is a specific

group focused on detection technology. The second committee will focus on current technologies and ways to improve in the future. One of the phases will examine current UXO remediation, active range UXO clearance and explosive ordnance disposal efforts. Hence, the UXO problem is serious and a highly visible issue within the DOD. The current approach to mine and UXO clearing is dangerous and labor intensive [Ref. 6]. In a typical UXO clearing scenario, Explosive Ordnance Disposal (EOD) technicians walk slowly and carefully over a contaminated field in an attempt to identify the presence of UXO's that may be fully buried, half buried, or totally on the surface. UXO's found on the surface are visually examined to determine their types and fuse mechanisms. If fuse mechanisms are recognized and the condition of the UXO permits (e.g., not rusted, decayed, or encased in soil), an effort is made to defuse UXO's. UXO's that cannot be defused are gathered at a safe location and are destroyed using shaped charges. Transporting any live UXO is extremely dangerous and the motion of the transport vehicle must be gentle. Moreover, buried UXO's must be unearthed first. Therefore, UXO clearing consists of detection, identification, defusing, excavating, transporting, and disposal. Through robotics and the use of advanced technology it should be possible to make UXO clearing tasks safer, cost-effective, and more efficient.

At the Naval Postgraduate School a team has been put together to develop a semiautonomous vehicle or robot, which will survey possible contaminated areas for UXO's. The "rotary" class vehicle by its very design is capable of independent driving and steering with each wheel. Hence, rotary vehicles are capable of stronger torque and traction than most other vehicles on rugged terrains. Rotary vehicles with two wheels (on a very smooth surface) have been shown to possess the aforementioned enhanced torque and traction – with an increase in the number of wheels on the vehicle the capabilities are significantly improved. The vehicle needs to be highly mobile and capable of producing very fine motion to negotiate and search contaminated sites. To fulfill the requirement of the special wheel architecture, the semiautonomous vehicle, called Shepherd (a rotary vehicle), is presently under development. Shepherd possesses stronger torque and traction on rugged terrain, because of its special architecture (i.e., useful for the

mine/UXO. Mission). The name "Shepherd" was given to the vehicle because of the protective function of the "shepherd" in many ancient cultures—hopefully this Shepherd will also save lives. Shepherd (Figure 1.1) is a four-wheeled vehicle with independent steering and driving capabilities. The four-wheel independent driving and steering capability provides Shepherd a high level of mobility and preciseness in motion control [Ref. 7].

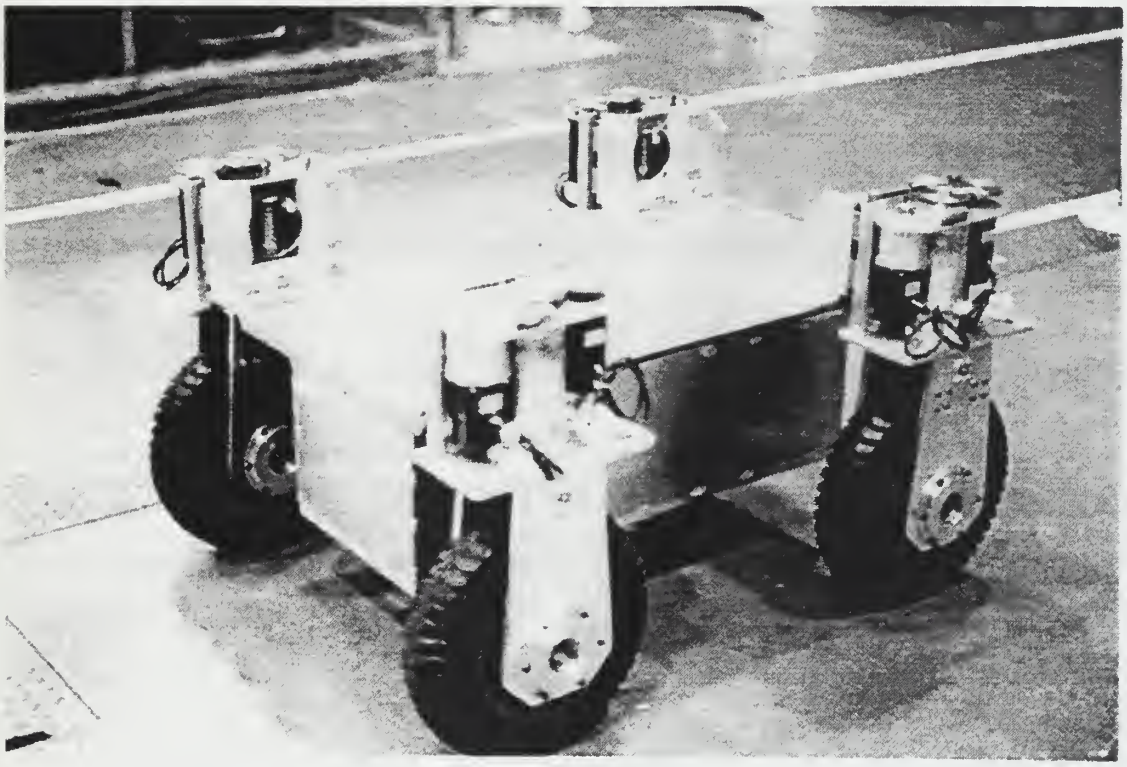

Figure 1.1: A "rotary" vehicle: Shepherd.

The fundamental objective of this thesis research project is to assist in the construction of a user-friendly and high-precision semiautonomous robotics tool to help the unexploded ordnance (UXO) and mine clearing mission.

Use Windows 95 HyperTerminal Program for Direct Connection

17. You are now back at the Windows desk top. Now ***double click*** on the ***ses2download*** shortcut to open a hard-line, under Windows 95 HyperTerminal (See figure 7).

Figure 7: **Windows ses2download Shortcut**

18. The next window to appear will be the open ***HyperTerminal window*** (See Figure 8). Press the "***reset***" button on the ***OMNIBYTE, Taurus board***. The Taurus bug (debugger) prompt will appear in the ***HyperTerminal window*** (See Figure 8).

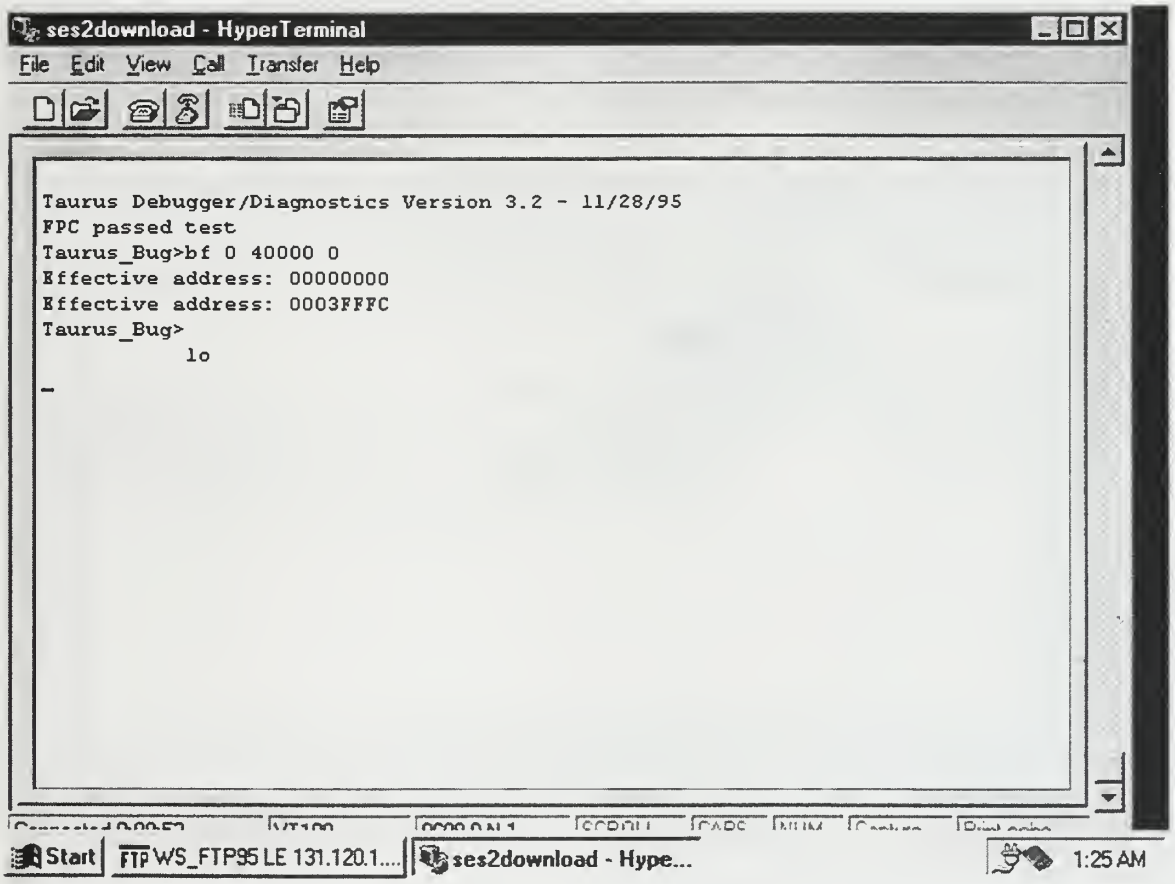

Figure 8: **Windows HypertTerminal Window**

19. Ensure the ***lever*** on the ***switch box*** is placed on ***console*** (this allows the console to emulate a VT220).

20. At the Taurus bug prompt type "***bf 0 40000 0***" and ***press the enter key***. This command is called block fill by the debugger it allows you to disable the parity error interrupt (***PEI***) and prevents problems caused by uninitialized variables. See Figure 8.

21. At the Taurus bug prompt type "***lo***" and ***press the enter key***. The "***lo***" command initiates the download from the console. See Figure 8.

22. Next place the ***lever*** on the ***switch box*** is placed on ***host*** (this makes possible the use of the RS232 protocol to download **shepherd.TXT from** c:\shepherdump to the Taurus board).

23. ***Click*** on the HyperTerminal "***Transfer "option*** and choose the "***Send Text File***". All the "***Send Text File***" parameters have been previously set, so there is no action to take in that regard.
See Figure 9.

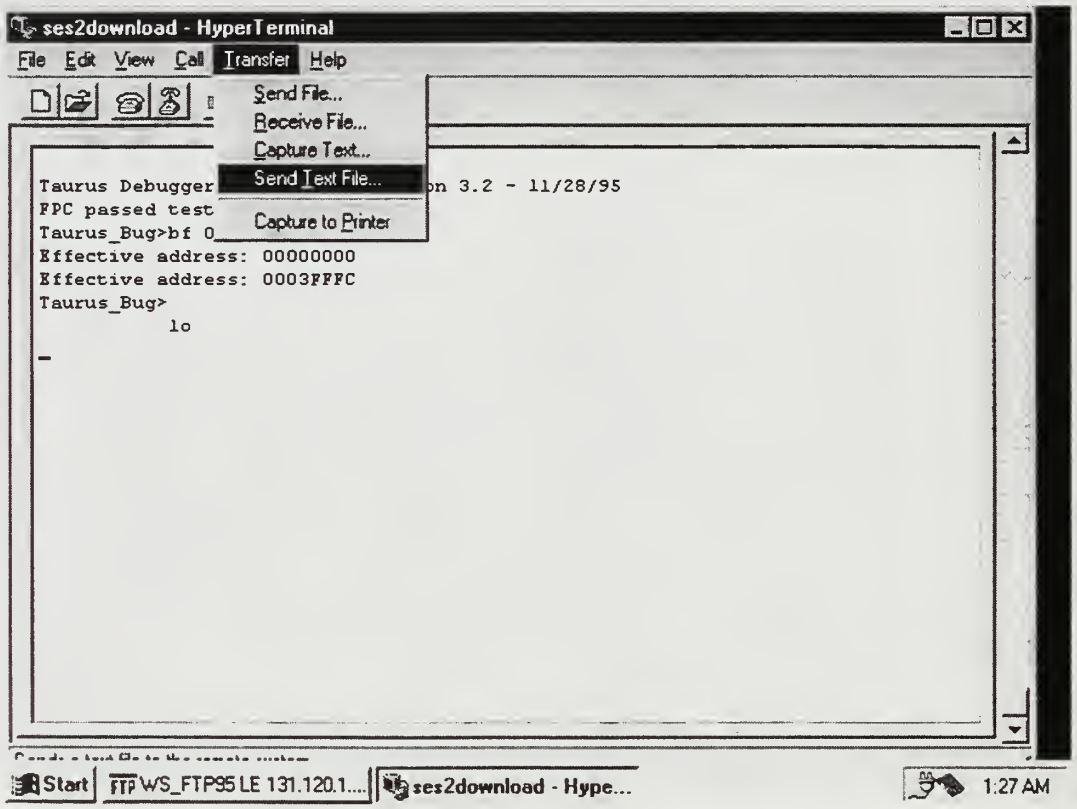

Figure 9: **Windows HyperTerminal Window**

24. Now move to the root directory and ***select the c:\shepherdump directory***, and double click on **shepherd.TXT** file. See figure 10.

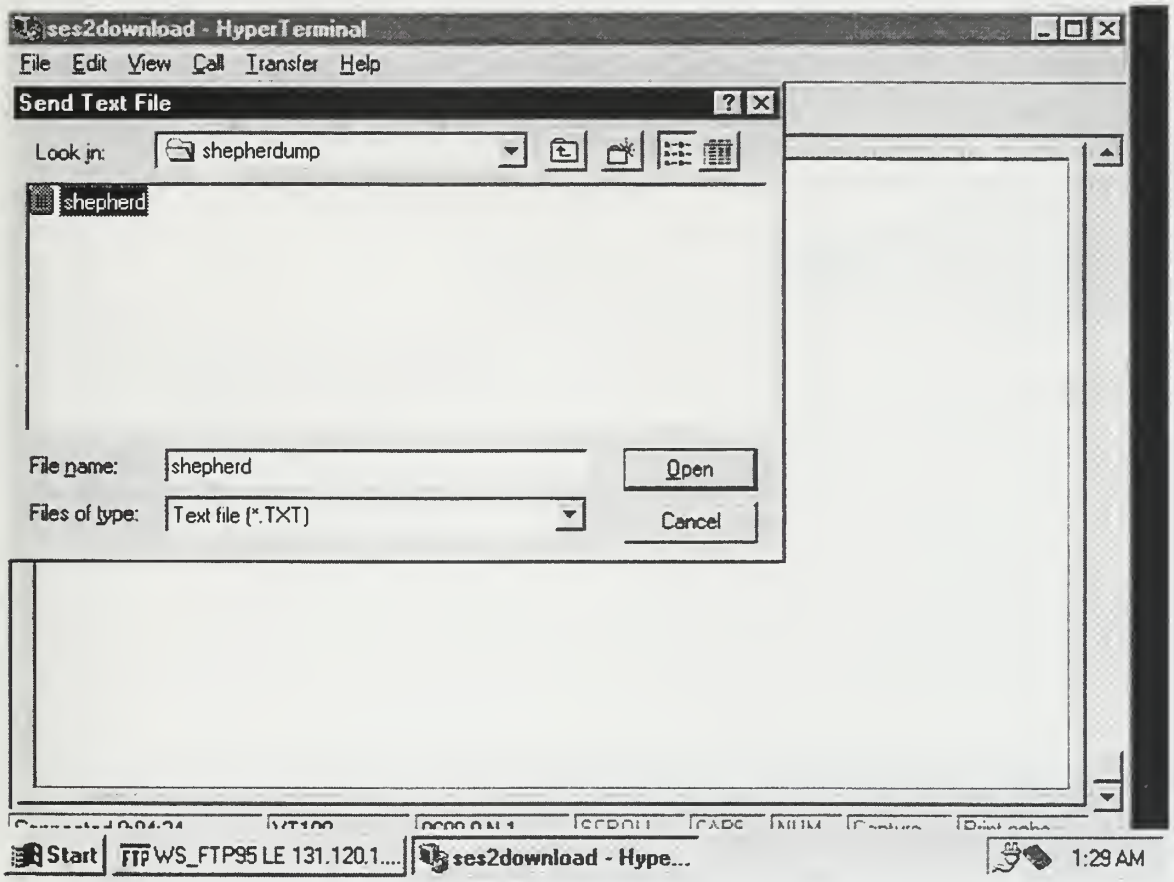

Figure 10: **Send File From shepherdump**

The download process is in motion. The "***red***" transmit light on the RS232 connector to the ***switch box*** will become faint while transmission is in progress. Once the transmission is complete the "***red***" transmit light on the RS232 connector to the ***switch box*** will become a constant red; the ***Hyperterminal window*** will pause during the transmission process. The ***Taurus bug prompt will appear in the HyperTerminal window*** after the transmission is complete.

25. Ensure the ***lever*** on the ***switch box*** is placed on ***console*** (this allows the console to emulate a VT220).

25. Now type ***"go"*** at the ***Taurus bug prompt*** and ***press the enter key*** (see figure 11). The program that you have previously downloaded will be executed.

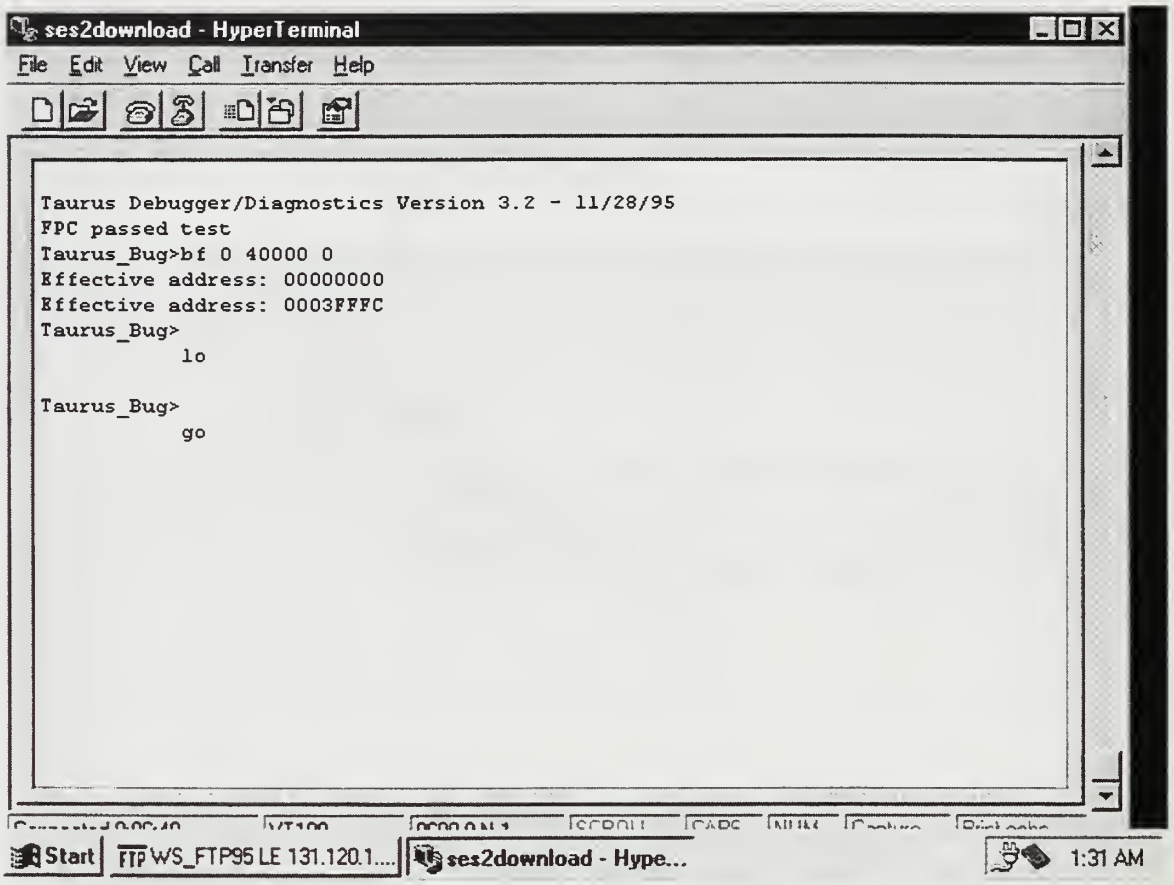

Figure 11: **Taurus Bug Prompt Returns After Transmisssion Completion and the "go" Command is given to Execute the Program.**

After the *"go"* has been given and the execution begins the Shepherd Main Menu appears for your selection.

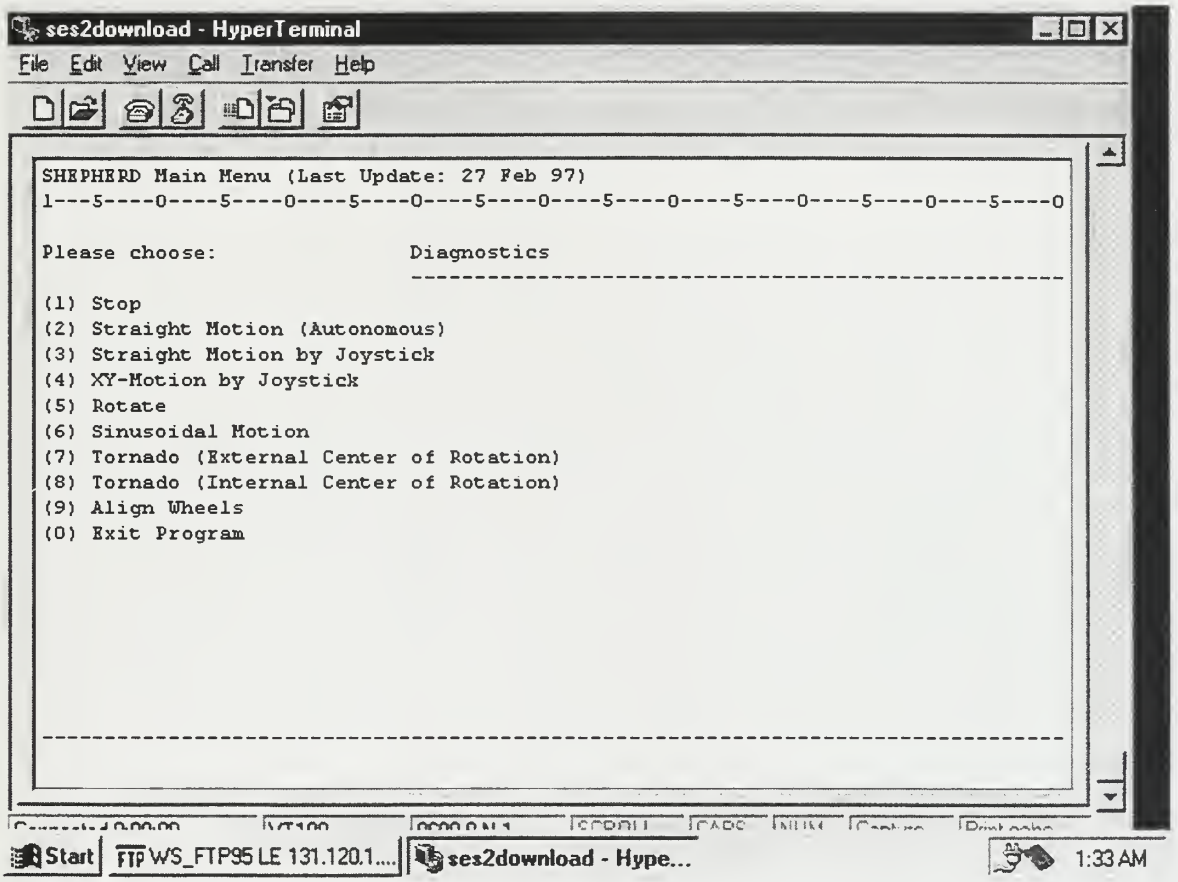

Figure 12: **Shepherd Main Menu**

Note: remember this is a ***quick guide*** and does not provided answers to questions concerning file size, debugger commands, and other requirements or constraints.

APPENDIX L: SENSING SIMULATION DATA

Distance	x	y	theta	psi
0.000	0.400	0.000	0.001	2.355
0.400	0.800	0.001	0.002	2.353
0.800	1.200	0.002	0.005	2.352
1.200	1.600	0.005	0.008	2.350
1.600	2.000	0.008	0.011	2.348
2.000	2.400	0.014	0.015	2.347
2.400	2.800	0.021	0.020	2.345
2.800	3.200	0.029	0.025	2.344
3.200	3.600	0.041	0.031	2.342
3.600	3.999	0.054	0.037	2.341
4.000	4.399	0.070	0.043	2.339
4.400	4.799	0.088	0.050	2.337
4.800	5.198	0.110	0.057	2.336
5.200	5.597	0.134	0.064	2.334
5.600	5.996	0.161	0.072	2.333
6.000	6.395	0.192	0.080	2.331
6.400	6.794	0.225	0.088	2.330
6.800	7.192	0.262	0.096	2.328
7.200	7.590	0.302	0.105	2.326
7.600	7.988	0.346	0.114	2.325
8.000	8.385	0.393	0.123	2.323
8.400	8.782	0.444	0.131	2.322
8.800	9.178	0.498	0.141	2.320
9.200	9.574	0.556	0.150	2.319
9.600	9.969	0.617	0.159	2.317

10.000	10.364	0.682	0.168	2.315
10.400	10.758	0.751	0.177	2.314
10.800	11.151	0.823	0.187	2.312
11.200	11.544	0.899	0.196	2.311
11.600	11.936	0.979	0.205	2.309
12.000	12.327	1.062	0.214	2.308
12.400	12.718	1.149	0.224	2.306
12.800	13.107	1.240	0.233	2.304
13.200	13.496	1.334	0.242	2.303
13.600	13.884	1.431	0.251	2.301
14.000	14.271	1.532	0.260	2.300
14.400	14.657	1.637	0.269	2.298
14.800	15.042	1.745	0.278	2.297
15.200	15.426	1.856	0.286	2.295
15.600	15.810	1.971	0.295	2.293
16.000	16.192	2.089	0.303	2.292
16.400	16.573	2.210	0.312	2.290
16.800	16.953	2.334	0.320	2.289
17.200	17.332	2.461	0.328	2.287
17.600	17.711	2.592	0.336	2.286
18.000	18.088	2.725	0.344	2.284
18.400	18.464	2.862	0.352	2.282
18.800	18.839	3.001	0.359	2.281
19.200	19.213	3.143	0.367	2.279
19.600	19.585	3.288	0.374	2.278
20.000	19.957	3.435	0.381	2.276
20.400	20.328	3.586	0.388	2.275
20.800	20.698	3.738	0.395	2.273
21.200	21.066	3.893	0.402	2.271

21.600	21.434	4.051	0.408	2.270
22.000	21.801	4.211	0.415	2.268
22.400	22.166	4.373	0.421	2.267
22.800	22.531	4.538	0.427	2.265
23.200	22.894	4.704	0.433	2.264
23.600	23.257	4.873	0.438	2.262
24.000	23.619	5.044	0.444	2.260
24.400	23.980	5.217	0.449	2.259
24.800	24.339	5.391	0.454	2.257
25.200	24.698	5.568	0.459	2.256
25.600	25.056	5.746	0.464	2.254
26.000	25.414	5.926	0.469	2.253
26.400	25.770	6.108	0.474	2.251
26.800	26.126	6.291	0.478	2.249
27.200	26.480	6.476	0.482	2.248
27.600	26.834	6.662	0.486	2.246
28.000	27.188	6.849	0.490	2.245
28.400	27.540	7.038	0.494	2.243
28.800	27.892	7.229	0.497	2.242
29.200	28.243	7.420	0.501	2.240
29.600	28.594	7.613	0.504	2.238
30.000	28.944	7.807	0.507	2.237
30.400	29.293	8.001	0.510	2.235
30.800	29.642	8.197	0.513	2.234
31.200	29.990	8.394	0.516	2.232
31.600	30.338	8.592	0.518	2.231
32.000	30.685	8.790	0.521	2.229
32.400	31.032	8.990	0.523	2.227
32.800	31.378	9.190	0.525	2.226

33.200	31.724	9.390	0.527	2.224
33.600	32.070	9.592	0.529	2.223
34.000	32.415	9.794	0.530	2.221
34.400	32.760	9.996	0.532	2.220
34.800	33.105	10.200	0.533	2.218
35.200	33.449	10.403	0.535	2.216
35.600	33.793	10.607	0.536	2.215
36.000	34.137	10.811	0.537	2.213
36.400	34.481	11.016	0.538	2.212
36.800	34.824	11.221	0.538	2.210
37.200	35.167	11.426	0.539	2.209
37.600	35.511	11.632	0.540	2.207
38.000	35.854	11.837	0.540	2.205
38.400	36.197	12.043	0.540	2.204
38.800	36.540	12.249	0.541	2.202
39.200	36.883	12.455	0.541	2.201
39.600	37.225	12.661	0.541	2.199
40.000	37.568	12.867	0.541	2.198
40.400	37.911	13.073	0.541	2.196
40.800	38.254	13.279	0.540	2.194
41.200	38.597	13.484	0.540	2.193
41.600	38.941	13.690	0.539	2.191
42.000	39.284	13.895	0.539	2.190
42.400	39.627	14.100	0.538	2.188
42.800	39.971	14.305	0.537	2.187
43.200	40.315	14.510	0.537	2.185
43.600	40.658	14.714	0.536	2.183
44.000	41.002	14.918	0.535	2.182
44.400	41.347	15.122	0.534	2.180

44.800	41.691	15.325	0.532	2.179
45.200	42.036	15.528	0.531	2.177
45.600	42.381	15.730	0.530	2.176
46.000	42.726	15.932	0.528	2.174
46.400	43.072	16.133	0.527	2.172
46.800	43.418	16.334	0.525	2.171
47.200	43.764	16.535	0.524	2.169
47.600	44.111	16.734	0.522	2.168
48.000	44.457	16.934	0.521	2.166
48.400	44.805	17.132	0.519	2.165
48.800	45.152	17.330	0.517	2.163
49.200	45.500	17.528	0.515	2.161
49.600	45.848	17.724	0.513	2.160
50.000	46.197	17.920	0.511	2.158
50.400	46.546	18.116	0.509	2.157
50.800	46.896	18.310	0.507	2.155
51.200	47.246	18.504	0.505	2.154
51.600	47.596	18.697	0.503	2.152
52.000	47.947	18.889	0.500	2.150
52.400	48.298	19.081	0.498	2.149
52.800	48.649	19.272	0.496	2.147
53.200	49.002	19.461	0.493	2.146
53.600	49.354	19.650	0.491	2.144
54.000	49.707	19.839	0.489	2.143
54.400	50.060	20.026	0.486	2.141
54.800	50.414	20.212	0.484	2.139
55.200	50.769	20.398	0.481	2.138
55.600	51.124	20.582	0.478	2.136
56.000	51.479	20.766	0.476	2.135

56.400	51.835	20.949	0.473	2.133
56.800	52.191	21.131	0.470	2.132
57.200	52.548	21.311	0.468	2.130
57.600	52.905	21.491	0.465	2.128
58.000	53.263	21.670	0.462	2.127
58.400	53.621	21.848	0.460	2.125
58.800	53.980	22.025	0.457	2.124
59.200	54.339	22.201	0.454	2.122
59.600	54.699	22.376	0.451	2.121
60.000	55.059	22.550	0.448	2.119
60.400	55.420	22.723	0.445	2.117
60.800	55.781	22.894	0.442	2.116
61.200	56.143	23.065	0.440	2.114
61.600	56.505	23.235	0.437	2.113
62.000	56.868	23.403	0.434	2.111
62.400	57.231	23.571	0.431	2.110
62.800	57.595	23.738	0.428	2.108
63.200	57.959	23.903	0.425	2.106
63.600	58.323	24.067	0.422	2.105
64.000	58.689	24.231	0.419	2.103
64.400	59.054	24.393	0.416	2.102
64.800	59.420	24.554	0.413	2.100
65.200	59.787	24.714	0.410	2.099
65.600	60.154	24.873	0.407	2.097
66.000	60.522	25.030	0.404	2.095
66.400	60.890	25.187	0.401	2.094
66.800	61.258	25.343	0.398	2.092
67.200	61.627	25.497	0.395	2.091
67.600	61.997	25.650	0.392	2.089

68.000	62.367	25.802	0.389	2.088
68.400	62.737	25.953	0.386	2.086
68.800	63.108	26.103	0.383	2.084
69.200	63.479	26.252	0.380	2.083
69.600	63.851	26.400	0.377	2.081
70.000	64.223	26.546	0.374	2.080
70.400	64.596	26.692	0.371	2.078
70.800	64.969	26.836	0.368	2.077
71.200	65.342	26.979	0.365	2.075
71.600	65.716	27.121	0.362	2.073
72.000	66.091	27.262	0.358	2.072
72.400	66.465	27.402	0.355	2.070
72.800	66.841	27.541	0.352	2.069
73.200	67.216	27.678	0.349	2.067
73.600	67.592	27.815	0.347	2.066
74.000	67.969	27.950	0.344	2.064
74.400	68.346	28.084	0.341	2.062
74.800	68.723	28.217	0.338	2.061
75.200	69.100	28.349	0.335	2.059
75.600	69.478	28.480	0.332	2.058
76.000	69.857	28.610	0.329	2.056
76.400	70.236	28.738	0.326	2.055
76.800	70.615	28.866	0.323	2.053
77.200	70.994	28.992	0.320	2.051
77.600	71.374	29.117	0.317	2.050
78.000	71.754	29.241	0.314	2.048
78.400	72.135	29.365	0.311	2.047
78.800	72.516	29.487	0.308	2.045
79.200	72.897	29.607	0.306	2.044

79.600	73.279	29.727	0.303	2.042
80.000	73.661	29.846	0.300	2.040
80.400	74.043	29.964	0.297	2.039
80.800	74.426	30.080	0.294	2.037
81.200	74.809	30.196	0.292	2.036
81.600	75.192	30.310	0.289	2.034
82.000	75.576	30.424	0.286	2.033
82.400	75.959	30.536	0.283	2.031
82.800	76.344	30.647	0.281	2.029
83.200	76.728	30.757	0.278	2.028
83.600	77.113	30.867	0.275	2.026
84.000	77.498	30.975	0.272	2.025
84.400	77.883	31.082	0.270	2.023
84.800	78.269	31.188	0.267	2.022
85.200	78.655	31.293	0.264	2.020
85.600	79.041	31.397	0.262	2.018
86.000	79.428	31.500	0.259	2.017
86.400	79.815	31.602	0.257	2.015
86.800	80.202	31.703	0.254	2.014
87.200	80.589	31.803	0.251	2.012
87.600	80.976	31.902	0.249	2.011
88.000	81.364	32.000	0.246	2.009
88.400	81.752	32.097	0.244	2.008
88.800	82.141	32.193	0.241	2.006
89.200	82.529	32.289	0.239	2.004
89.600	82.918	32.383	0.236	2.003
90.000	83.307	32.476	0.234	2.001
90.400	83.696	32.568	0.232	2.000
90.800	84.085	32.659	0.229	1.998

91.200	84.475	32.750	0.227	1.997
91.600	84.865	32.839	0.224	1.995
92.000	85.255	32.928	0.222	1.993
92.400	85.645	33.015	0.220	1.992
92.800	86.036	33.102	0.217	1.990
93.200	86.427	33.188	0.215	1.989
93.600	86.817	33.273	0.213	1.987
94.000	87.209	33.357	0.210	1.986
94.400	87.600	33.440	0.208	1.984
94.800	87.991	33.522	0.206	1.982
95.200	88.383	33.603	0.204	1.981
95.600	88.775	33.684	0.201	1.979
96.000	89.167	33.763	0.199	1.978
96.400	89.559	33.842	0.197	1.976
96.800	89.951	33.920	0.195	1.975
97.200	90.344	33.997	0.193	1.973
97.600	90.736	34.073	0.191	1.971
98.000	91.129	34.149	0.189	1.970
98.400	91.522	34.223	0.186	1.968
98.800	91.915	34.297	0.184	1.967
99.200	92.309	34.370	0.182	1.965
99.600	92.702	34.442	0.180	1.964
100.000	93.096	34.513	0.178	1.962
100.400	93.489	34.584	0.176	1.960
100.800	93.883	34.654	0.174	1.959
101.200	94.277	34.723	0.172	1.957
101.600	94.671	34.791	0.170	1.956
102.000	95.066	34.858	0.168	1.954
102.400	95.460	34.925	0.166	1.953

102.800	95.855	34.991	0.165	1.951
103.200	96.249	35.056	0.163	1.949
103.600	96.644	35.120	0.161	1.948
104.000	97.039	35.184	0.159	1.946
104.400	97.434	35.247	0.157	1.945
104.800	97.829	35.309	0.155	1.943
105.200	98.224	35.371	0.154	1.942
105.600	98.620	35.432	0.152	1.940
106.000	99.015	35.492	0.150	1.938
106.400	99.411	35.551	0.148	1.937
106.800	99.806	35.610	0.146	1.935
107.200	100.202	35.668	0.145	1.934
107.600	100.598	35.725	0.143	1.932
108.000	100.994	35.782	0.141	1.931
108.400	101.390	35.838	0.140	1.929
108.800	101.786	35.893	0.138	1.927
109.200	102.182	35.948	0.136	1.926
109.600	102.579	36.002	0.135	1.924
110.000	102.975	36.056	0.133	1.923
110.400	103.372	36.108	0.132	1.921
110.800	103.768	36.161	0.130	1.920
111.200	104.165	36.212	0.128	1.918
111.600	104.562	36.263	0.127	1.916
112.000	104.959	36.313	0.125	1.915
112.400	105.355	36.363	0.124	1.913
112.800	105.752	36.412	0.122	1.912
113.200	106.149	36.461	0.121	1.910
113.600	106.547	36.509	0.119	1.909
114.000	106.944	36.556	0.118	1.907

114.400	107.341	36.603	0.116	1.905
114.800	107.738	36.649	0.115	1.904
115.200	108.136	36.695	0.114	1.902
115.600	108.533	36.740	0.112	1.901
116.000	108.931	36.784	0.111	1.899
116.400	109.328	36.828	0.109	1.898
116.800	109.726	36.872	0.108	1.896
117.200	110.124	36.915	0.107	1.894
117.600	110.521	36.957	0.105	1.893
118.000	110.919	36.999	0.104	1.891
118.400	111.317	37.040	0.103	1.890
118.800	111.715	37.081	0.102	1.888
119.200	112.113	37.121	0.100	1.887
119.600	112.511	37.161	0.099	1.885
120.000	112.909	37.200	0.098	1.883
120.400	113.307	37.239	0.097	1.882
120.800	113.705	37.277	0.095	1.880
121.200	114.103	37.315	0.094	1.879
121.600	114.502	37.353	0.093	1.877
122.000	114.900	37.389	0.092	1.876
122.400	115.298	37.426	0.091	1.874
122.800	115.697	37.462	0.089	1.872
123.200	116.095	37.497	0.088	1.871
123.600	116.494	37.532	0.087	1.869
124.000	116.892	37.567	0.086	1.868
124.400	117.291	37.601	0.085	1.866
124.800	117.689	37.635	0.084	1.865
125.200	118.088	37.668	0.083	1.863
125.600	118.486	37.701	0.082	1.861

APPENDIX M: SENSING SIMULATION CODE (MAIN2.CC)

```
// File: main2.cc

// Name: Edward Mays

// Sensing Simulation

// Unix

// GCC

// Date: 26 August 1997

//

// Description

// THIS PROGM SIMULATES THE MOVEMENT OF A SQARE OBJECT ALONG A
//PATH.  THE OBJECT'S PATH DIRECTION (THETA) IS CHANGING, AS IS THE
//OBJECTS ORIENTATION (PSI).  LINE TRACKING IS USED AND THE X-AXIS IS
THE //REFERENCE LINE.  THE REFERENCE LINE IS INCREMENTED BY 40
UNITS IN THE

// ---------------------------------

// Header file info

// ---------------------------------

#include <iostream.h>

#include <math.h>

#include <fstream.h>

#include <stdio.h>



#define PI     3.14159265358979323846

#define RAD    57.29577951308232087684
```

```
double deltaTime = 0.01;// 0.01
double Vel = 40.0;
double omega = -0.1570796327;

FILE *f0, *f1, *f2, *f3, *f4, *f5,*f6 ;      //PTR TO FILE FOR OUTPUT DATA


//structure to hold configuration including x, y, theta, and kappa

typedef struct{
   double x;
   double y;   }
  POINT;

  typedef struct{
   POINT Point;
  double Theta;
  double Kappa;
   double Psi;
   }
  CONFIGURATION;


  // ---------------------------------
  //Function: GetSmooth
  //Return Value:n/a
```

```
//Parameters:  function parm  list
//Purpose: gets users input for s0/smoothness
// ----------------------------------
double GetSmooth(double &s0)
{



cout << "enter your value for smoothness (negatives not allowed)" <<endl;
 cin >> s0;
 return (s0 >= 0.0);

}// GetSmooth


// ----------------------------------
//Function: InitConfig
//Return Value:n/a
//Parameters:  function parm  list
//Purpose:  SETS INITIAL CONFIGURATION
// ----------------------------------
void InitConfig(CONFIGURATION& q_init, CONFIGURATION&
q_xaxis,CONFIGURATION& qbody,CONFIGURATION& qfrontR,
    CONFIGURATION& qfrontL, CONFIGURATION& qrearR, CONFIGURATION&
qrearL, CONFIGURATION& qsnapshot,
        double &s0, double &deltaS)
{
cout<<"Setting the initial configuration"
```

```
<<"x=0, y = 0, theta = 0, and kappa = 0 "<<endl;

q_init.Point.x = 0.0;
q_init.Point.y = 0.0;
q_init.Theta   = 0.0;
q_init.Kappa   = 0.0;
q_init.Psi   = 2.356219449; /* 3*PI/4.0 */

cout<<"Setting the reference line configuration"
<<"x = 0, y = 40, theta = 0 , and kappa = 0 "<<endl;

q_xaxis.Point.x  = 0.0;
q_xaxis.Point.y  = 40.0;
q_xaxis.Theta    = 0.0;
q_xaxis.Kappa    = 0.0;


 //individual wheels
qfrontR.Point.x = 40;   /* wheel1 */
qfrontR.Point.y = -40;

qfrontL.Point.x = 40;   /* wheel2 */
qfrontL.Point.y = 40;

qrearR.Point.x = -40;   /* wheel3 */
qrearR.Point.y = -40;
```

```
qrearL.Point.x = -40;   /* wheel 4 */
qrearL.Point.y =  40;

qsnapshot.Point.x = 0.0;
qsnapshot.Point.y = 0.0;

cout<<"Enter size constant for smoothness <return>"<<endl;
GetSmooth(s0);

cout<<"Entering Step size constant deltaS(deltaS=Vel*deltaT)."<<endl;
deltaS = Vel*deltaTime;//.05 orig


}// InitConfig



// ---------------------------------
//Function: CreateConst
//Return Value:n/a
//Parameters:  function parm  list
//Purpose:  create constants for
//         steering function dk/ds
// ---------------------------------
void CreateConst(double &a, double &b, double &c, double &s0)
```

```
{
double k;
 k = 1.0/s0;  //all consts by def, including curvature
 a = 3.0*k;
 b = 3.0*k*k;
c = k*k*k;
}// CreateConst



// ----------------------------------
//Function: GetSteerL
//Return Value:n/a
//Parameters:  function parm  list
//Purpose:     lambda=dk/ds  (LINEAR STEERING FUNCTION)
// ----------------------------------

double GetSteerL(double &a, double &b, double &c, CONFIGURATION& q,
    CONFIGURATION& q_xaxis)
{
```

```
double delta_r;

delta_r = -(q.Point.x - q_xaxis.Point.x)*sin(q_xaxis.Theta) +
          (q.Point.y - q_xaxis.Point.y)*cos(q_xaxis.Theta);

return (-(a*q.Kappa + b*(q.Theta -  q_xaxis.Theta) + c*delta_r));
} //GetSteerL



// ----------------------------------
//Function: GetDeltakappa
//Return Value:n/a
//Parameters:  function parm  list
//Purpose:   DETERMINES THE KAPPA DIFFERENCE PER INCREMENT OF S
// ----------------------------------

double GetDeltakappa(double &Dk_Ds, double &deltaS, double &deltakappa)
{
deltakappa = Dk_Ds*deltaS;
return(deltakappa);

}//GetDeltakappa
```

```
// ---------------------------------
//Function: returnkappa
//Return Value:n/a
//Parameters:  function parm  list
//Purpose:  CALCULATES NEW VALUE FOR KAPPA USING deltaK
// ---------------------------------

CONFIGURATION returnkappa(double &deltakappa, CONFIGURATION &q)
{
 q.Kappa = q.Kappa + deltakappa;
 return q;

}//returnkappa



// ---------------------------------
//Function: GetS
//Return Value:n/a
//Parameters:  function parm  list
//Purpose:  INCREMENTS S THROUGH EACH ITERATION OF THE WHILE LOOP
// ---------------------------------

double GetS(double &s, double &deltaS)
{
```

```
  s = s + deltaS;
return s;

}//GetS

// ----------------------------------
//Function: GetDeltaTheta
//Return Value:n/a
//Parameters:  function parm  list
//Purpose: COMPUTES CHANGE IN THETA PER INCREMENT OF S
// ----------------------------------

double GetDeltaTheta(CONFIGURATION &q, double &deltaS, double &deltaT)
{
deltaT =  q.Kappa*deltaS;
return(deltaT);

}//GetDeltaTheta

// ----------------------------------
//Function: Circ
//Return Value:n/a
//Parameters:  function parm  list
//Purpose:  Circ function from notes 6.29
// ----------------------------------
```

```
void Circ(double Length, double alpha, CONFIGURATION &q)
{
double alpha2, alpha4;
 alpha2=alpha*alpha;
alpha4=alpha2*alpha2;



//configuration q1
q.Point.x = (1.0 - alpha2/6.0 + alpha4/120.0)*Length;
q.Point.y = (0.5 - alpha2/24.0 + alpha4/720.0)*Length*alpha;
q.Theta  = alpha;



}//Circ



// ---------------------------------
//Function: Compose
//Return Value:n/a
//Parameters:  function parm  list
//Purpose: updates the configuration and computes new config (notes 6.2)
// ---------------------------------
CONFIGURATION Compose(CONFIGURATION& q1,CONFIGURATION&
q2,CONFIGURATION& q3, double& s,double& deltaTime)
```

```
{   double x,y,
sinTheta = sin(q1.Theta),
cosTheta = cos(q1.Theta);

x = q1.Point.x + q2.Point.x*cosTheta - q2.Point.y*sinTheta;
y = q1.Point.y + q2.Point.x*sinTheta + q2.Point.y*cosTheta;
q3.Point.x = x;
q3.Point.y = y;
q3.Theta = q1.Theta + q2.Theta;

 q3.Psi = q1.Psi + (omega * deltaTime);   /* how to handle move left/right? */

 fprintf(f6,"%10.3f %10.3f %10.3f %10.3f %10.3f\n",
         s,q3.Point.x, q3.Point.y,q3.Theta, q3.Psi);
return q3;

}// end Compose



CONFIGURATION Compose2(CONFIGURATION& q1,CONFIGURATION& q2,
CONFIGURATION& q3) /*position */
{   double x,y,
 sinTheta = sin(q1.Psi),
cosTheta = cos(q1.Psi);
```

```
 x = q1.Point.x + q2.Point.x*cosTheta - q2.Point.y*sinTheta;
y = q1.Point.y + q2.Point.x*sinTheta + q2.Point.y*cosTheta;

q3.Point.x = x;
q3.Point.y = y;
 return q3;
}// end Compose2


// ---------------------------------
//Function: Openfile
//Return Value:n/a
//Parameters:  function parm  list
//Purpose: To compute transposition
// ---------------------------------
void Openfile()
{
f0 = fopen("drk.dat","w");
 f1 = fopen("wheel1.dat","w");
 f2 = fopen("wheel2.dat","w");
 f3 = fopen("wheel3.dat","w");
f4 = fopen("wheel4.dat","w");
 f5 = fopen("composite.dat","w");
f6 = fopen("psi.dat","w");
}// Openfile
```

```
// ----------------------------------
//Function: Print
//Return Value:n/a
//Parameters:  function parm  list
//Purpose: To compute transposition
// ----------------------------------
void printFile(FILE *f,CONFIGURATION &q)
{

 fprintf(f,"%10.3f %10.3f\n",
          q.Point.x, q.Point.y);



}// printFile


// ----------------------------------
//Function: blankLine
//Return Value:n/a
//Parameters:  function parm  list
//Purpose: To compute transposition
// ----------------------------------
void blankLine(FILE *f)
{
```

```
fprintf(f,"\n");



}// blankLine




// ---------------------------------

//Function: updateWheels

//Return Value:n/a

//Parameters:  function parm  list

/Purpose:  create constants for

//          steering function dk/ds

// ---------------------------------

void updateWheels(CONFIGURATION& qbody, CONFIGURATION& qfrontR, CONFIGURATION& qfrontL, CONFIGURATION qrearR, CONFIGURATION qrearL, CONFIGURATION& qwheel1, CONFIGURATION& qwheel2, CONFIGURATION& qwheel3, CONFIGURATION& qwheel4, CONFIGURATION& q3,int& s2)

{    printFile(f0,qbody);


qwheel1 = Compose2(qbody,qfrontR,q3);

printFile(f1,qwheel1);

blankLine(f1);

qwheel2 = Compose2(qbody,qfrontL,q3);

printFile(f2,qwheel2);
```

```
blankLine(f2);
qwheel3 = Compose2(qbody, qrearR, q3);
printFile(f3,qwheel3);
blankLine(f3);
qwheel4 = Compose2(qbody, qrearL,q3);
printFile(f4,qwheel4);
blankLine(f4);

if((s2==0.0)||(s2%100==0)){

printFile(f5,qwheel1);
printFile(f5,qwheel2);
printFile(f5,qwheel4);
printFile(f5,qwheel3);
printFile(f5,qwheel1);
blankLine(f5);
}
}// updateWheels




int main()
```

```
{ CONFIGURATION q, q_xaxis, New_q, qbody, qfrontR, qfrontL, qrearR, qrearL,
qwheel1, qwheel2, qwheel3, qwheel4, qsnapshot,q3;


int ix,s2,counter;
double a, b, c;                    //constants equation 6.3
double s,s0;                       //s0 is smoothness, s is the incremental step
double Dk_Ds;
double deltaS;
double deltaT;
double deltaK;
const double Sdig = 0.001;         //const used for prec/toler
double smax=400.0;


InitConfig(q, q_xaxis,  qbody, qfrontR, qfrontL, qrearR, qrearL,qsnapshot,s0, deltaS);
//configuaration set up


Openfile();
printFile(f0,q);
printFile(f1,qfrontR);                    //write initial config to file
printFile(f2,qfrontL);
printFile(f3,qrearR);
 printFile(f4,qrearL);



//printFile(f5,qfrontR);
//printFile(f5,qfrontL);
```

```
//printFile(f5,qrearL);
//printFile(f5,qrearR);

printFile(f5,qfrontR);
blankLine(f5);

CreateConst(a, b, c, s0);                    //calcs consts
s = 0.0;
s2=s;
counter=0;
for(ix=0; ix<10;ix++){
do
{//

Dk_Ds = GetSteerL(a, b, c, q, q_xaxis);          //calculates lambda =dk/ds
deltaK = GetDeltakappa(Dk_Ds, deltaS, deltaK);   //lambda*deltaS
returnkappa(deltaK, q);                      //Kappa <= kappa + deltaK
deltaT = GetDeltaTheta(q, deltaS, deltaT);       //Theta <= Theta + deltaT
Circ(deltaS, deltaT, New_q);                   //cir
qbody = Compose(q, New_q, q, s, deltaTime);                  //compose
updateWheels(qbody,qfrontR, qfrontL, qrearR,
                qrearL,qwheel1, qwheel2, qwheel3,
                qwheel4,q3,s2);
GetS(s, deltaS);
s2=s;
```

```
}while (s<smax);
// }while ((fabs(q.Point.y) > Sdig)||(fabs(q.Theta) > Sdig) ||
//       (fabs(q.Kappa) > Sdig));
s=0.0;
s2=s;
q_xaxis.Point.y  = q_xaxis.Point.y + 40.0;

if(ix%2==0){
q_xaxis.Theta    = PI;
q.Theta    = PI;
omega = fabs(omega);
}else{
q_xaxis.Theta    = 0.0;
q.Theta    = 0.0;
omega = -omega;
}
}
fclose(f0);
fclose(f1);
fclose(f2);
fclose(f3);
fclose(f4);
fclose(f5);
fclose(f6);
return 0;
}//end main2.cc
```

APPENDIX N: INPUT VS. OUPUT VELOCITY

-1024 -87.657

-1023 -87.429

-1020 -86.975

-1015 -86.747

-1010 -86.406

-1005 -85.838

-1000 -85.383

-900 -76.856

-800 -67.998

-700 -59.082

-600 -51.048

-500 -42.748

-400 -34.107

-375 -31.834

-350 -29.673

-300 -25.580

-250 -21.260

-225 -19.100

-200 -17.053

-175 -14.780

-150 -12.506

-125 -10.459

-100 -8.413

-90 -7.503

-80 -6.707

-70 -5.798

-60 -5.002

-50 -4.092

-40 -3.297

-30 -2.387

-20 -1.591

-10 -.682

-5 -.341

-1 -.113

0.0 0.0

1023 87.429

1020 86.975

1015 86.747

1010 86.406

1005 85.838

1000 85.383

900 76.856

800 67.998

700 59.082

600 51.048

500 42.748

400 34.107

375 31.834

350 29.673

300 25.580

250 21.260

225 19.100

200 17.053

175 14.780

150 12.506

125 10.459

100 8.413

90 7.503

80 6.707

70 5.798

60 5.002

50 4.092

40 3.297

30 2.387

20 1.591

10 .682

5 .341

1 .113

APPENDIX O: INPUT VS OUTPUT STEERING RATES

A. DESIRED INPUT RATE VS ACTUAL (BOTH ESTIMATED AND SOFTWARE DEPENDENT)

Desired rate of turn	Time Stop watch (sec)	Estimated Rate (rad/s)	Software Measured Rate (rad/s, average)
1	6.0	1.00000	0.98174
2	3.5	1.79485	1.95667
3	2.19	2.86849	2.93160
5	1.69	2.71716	3.90653
5.5	No data	No data	4.88828
10	No data	No data	5.23598
20	No data	No data	5.23598
30	No data	No data	5.23598

Figure A.1 Inputs and results from massaged data (error). No data entries exist because the revolutions were too fast for hand timing.

B. DESIRED INPUT RATE VS OUPUT FOR EACH WHEEL (SOFTWARE DEPENDENT

Below M5, M6, M7, and M8 reperesent the steering motors for wheel 1, wheel 2, wheel3, and wheel4 respectively.

Desired Rate of turn	M5 Rate	M6 Rate	M7 Rate	M8 Rate
1	1.002	.9975	.9965	.999
2	2.0025	1.997	1.990	1.997
3	3.005	2.995	2.9975	3.005
4	4.004	3.996	3.9925	3.996
5	5.008	5.002	4.9985	5.002

Desired Rate of turn	M5 Rate	M6 Rate	M7 Rate	M8 Rate
5.1	5.1035	5.0935	5.093	5.0955
5.2	5.2065	5.198	5.1955	5.198
5.3	5.238	5.235	5.234	5.235
5.4	5.238	5.235	5.234	5.235
-1	-1.001	-1.002	-1.002	-1.002
-2	-2.001	-2.003	-2.003	-2.004
-3	-3.0055	-3.006	-3.0055	-3.006
-4	-4.007	-4.006	-4.0015	-4.008
-5	-5.010	-5.010	-5.009	-5.010
-5.1	-5.107	-5.110	-5.1035	-5.112
-5.2	-5.2105	-5.2125	-5.2075	-5.214
-5.3	-5.238	-5.2415	-5.237	-5.2535
-5.4	-5.238	-5.2415	-5.237	-5.2535

Figure A.2: Desired (commanded) rate of turn vs. actual "free floating" motor rate.

C. DESIRED INPUT (DIGIT MANIPULATION) VS OUTPUT RATE

Input (digits)	M5 Rate (rad/s)	M6 Rate (rad/s)	M7 Rate (rad/s)	M8 Rate (rad/s)
10	.056	.045	.044	.045
20	.102	.101	.100	.100
30	.152	.143	.143	.143
40	.205	.203	.203	.203
50	.261	.250	.248	.249
60	.306	.306	.305	.305
70	.363	.350	.349	.350
80	.409	.408	.407	.407
90	.466	.454	.452	.455
100	.511	.510	.510	.510
200	1.024	1.022	1.022	1.022

Input (digits)	M5 Rate (rad/s)	M6 Rate (rad/s)	M7 Rate (rad/s)	M8 Rate (rad/s)
300	1.539	1.533	1.533	1.533
400	2.049	2.046	2.045	2.045
500	2.561	2.556	2.556	2.556
600	3.074	3.068	3.067	3.068
700	3.584	3.579	3.579	3.579
800	4.097	4.092	4.091	4.092
900	4.610	4.602	4.602	4.602
1000	5.124	5.116	5.114	5.116
1010	5.174	5.172	5.170	5.172
1020	5.226	5.218	5.216	5.218
1021	5.233	5.226	5.222	5.225
1022	5.235	5.233	5.231	5.232
1023	5.237	5.235	5.234	5.235

Figure A.3: Desired (commanded) rate of turn vs. actual "free floating" motor rate for each wheel using input digits .

D. WHEEL 4 ROTATION DATA

	Clockwise Rotation	Counterclockwise Rotation
1	000.867	360.390
2	000.878	360.390
3	000.976	360.363
4	000.933	360.414
5	000.984	360.371
6	000.992	360.394
7	000.992	360.453
8	000.902	360.394
9	000.996	360.445
10	000.996	360.476
11	001.003	360.402

	Clockwise Rotation	Counterclockwise Rotation
12	000.917	360.417
13	000.996	360.433
14	001.003	360.468
15	000.972	360.468
16	000.863	360.472
17	000.968	360.398
18	000.941	360.425
19	000.957	360.480
20	000.937	360.480

Figure A.4: Wheel 4 data based on position of rest after direction of turn.

LIST OF REFERENCES

1. Review & Outlook, *The Land Mine Dilemma,* The Wall Street Journal, May 7, 1996, Page A18.

2. Tom Nasland and John Barry, *Buried Terror*, Newsweek Special Report, Newsweek, April 8, 1996, pp. 24-27.

3. Steven Ashley, *Searching for Land Mines*, Mechanical Engineering, April 1996, pp. 62-67.

4. X.K. Maruyama, *Technologies in Support of International Peace Operations – Military Technologies for Ground Forces,* Naval Postgraduate School, February 1996.

5. Proceedings of Autonomous Vehicles in Mine Countermeasures Symposium, Naval Postgraduate School, April 4-7, 1995.

6. Evaluation of the Individual Demonstrator performance at the Unexploded Ordnance Advanced Technology Demonstration Program at Jefferson Proving Ground (Phase I), Naval Explosive Ordnance Disposal Technology Division, 1995.

7. Kanayama, Y., *Introduction to Motion Planning lecture Notes of CS4313*, Department of Computer Science, Naval Postgraduate School, 1996.

8. Mitsubishi Heavy Industries, Takasago Research Labs, Rotary Vehicle: Wheel Design Structure Memo, December 1995, paragraph 3.2.

9. Mitsubishi Heavy Industries, Takasago Research Labs, Rotary Vehicle: Wheel Design Structure Memo, December 1995, Annex.

10. Stallings, W., *Computer Organization and Architecture*, Third Edition, Macmillan Publishing, New York, 1993, pp. 90-112.

11. Taurus 68040/68060 VMEbus Single Board Computer, User's Manual, Omnibyte Corporation, Chicago, Illinois, March 1995, pp. 2-1, 2-47, and 2-53.

12. Mays, E., *Shepherd Chronology and Update for CS4920: Seminar in Military Robotics*, Department of Computer Science, Naval Postgraduate School, May 1997.

13. Acromag Series 9210/9215, Analog Output Board, User's Manual, Acromag Incorporated, Wixom Michigan, January 1994, pp. 1.1-1.2.

14. Acromag Series 9421, VME Isolated Digital Input Board, User's Manual, Acromag Incorporated, Wixom Michigan, January 1994, pp. 1.1-1.2.

15. VMIVME-2170A, 32 Bit Optically Coupled Digital Output Board, Instructional Manual, VME Microsystems International Corporation, Huntsville Alabama, February 1994, page 4-2.

16. User Manual, IP-Quadrature, Four Channel Quadrature Decoder Industry Pack, Green Spring Computers, Menlo Park California, 1995, pp. 7-10 and 20-27.

17. Elgerd, O., I., Control Systems Theory, McGraw Hill, New York, 1967, pp. 183-186.

18. Ibid., p. 185.

19. Ibid., pp. 60 and 128.

20. OMNIBug (020Bug) Monitor/Debugger program, Omnibyte Corporation, Chicago, Illinois, March 1995, pp. 1-4, A1, and A3.

21. Taurus 68040/68060 VMEbus Single Board Computer, User's Manual, Omnibyte Corporation, Chicago, Illinois, March 1995, pp. 2-13 through 2-18.

22. Ibid., p. 2-16.

23. Kanayama, Y., *"Rotary Vehicle" That Moves with Three Degrees of Freedom*, Department of Computer Science, Naval Postgraduate School, 1996.

24. Kanayama, Y. and Yun, X., Rigid Body Motion Analysis towards Rotary Vehicle, Naval Postgraduate School, 1997.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center.....2
   8725 John J. Kingman Road, Ste 0944
   Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Rd.
   Monterey, CA 93943-5101

3. Chairman, Code CS 1
   Computer Science Department
   Naval Postgraduate School
   Monterey, CA 93943

4. Dr. Xiaoping Yun, Code EC/YX 1
   Electrical Engineering Department
   Naval Postgraduate School
   Monterey, CA 93943

5. Dr. Robert B. McGhee, Code CS/Mz 1
   Computer Science Department
   Naval Postgraduate School
   Monterey, CA 93943

6. Dr. Yutaka Kanayama, Code CS/Kz.....1
   Computer Science Department
   Naval Postgraduate School
   Monterey, CA 93943

7. Major Edward J. Mays 4
PM TRCS
Bldg 455 (Oceanport Ave)
ATTN: SFAE-C3S-TRC-EPLRS (Major Mays)
Fort Monmouth, N.J. 07703-5504
Inbound - 15 October 97

8. Lieutenant Ferdinand A. Reid.....4
Department Head Class 150
SWOSCOLCOM
446 Cushing Road
Newport, RI 02841-1209

9. Director, Training and Education 1
MCCDEC, Code C46
1019 Elliot Road
Quantico, VA 22134-5027

10. Director, Marine Corps Research Center 2
MCCDEC, Code C40RC
2040 Broadway Street
Quantico, VA 22134-5107

11. Director Studies and Analysis Division.....1
MCCDC, Code C45
3300 Russell Road
Quantico, VA 22134-5130

12. Marine Corps Representative.....1
Naval Postgraduate School
Code 037, Bldg. 234, HA-220
699 Dyer Road
Monterey, CA 93940

13. Marine Corps Tactical Systems Support Activity.....1
Technical Advisory Branch
Attn: Maj J. C. Cummiskey
Box 555171
Camp Pendleton, CA 92055-5080

9 483NPG
TH 2961
10/99 22527-100 MULE

DUDLEY KNOX LIBRARY
3 2768 00366905 2